

Deep Reinforcement Learning — and — Recurrent World Models Facilitate Policy Evolution

David Ha and Jurgen Schmidhuber

Presentation by Chris Larsen
NeAR_{Y_aDLL} Graduate Researcher

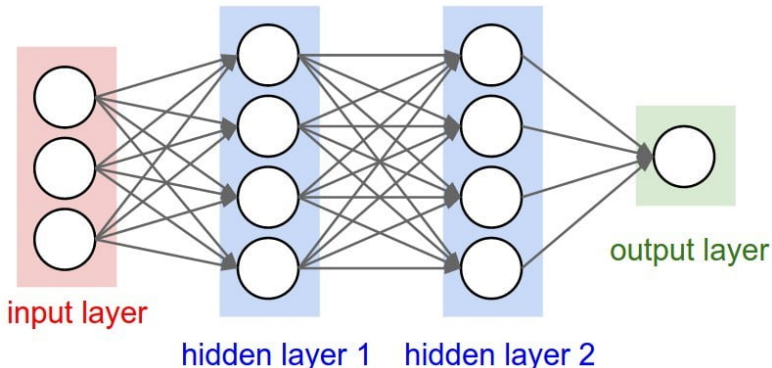
October 4, 2018

- Senses \Rightarrow Internal Model \Rightarrow Actions
- Each of us carries a predictive internal model inside our brain
 - We use this model even without full state knowledge
- To do this in RL, a RNN (a neural network through time) is used to represent the model
 - Captures temporal and spatial representations of data
- If the model is a sufficient representation of the environment, we can train from that model

- 1 Introduction
- 2 Deep Learning
- 3 Deep Reinforcement Learning
- 4 Agent Model
 - Visual Component
 - Model Component
 - Controller Component
- 5 Car Racing Experiment
- 6 VizDoom Experiment
 - τ
 - Cheating
- 7 Discussion

The Fastest Deep Learning Intro Ever

Deep learning is a sub field of machine learning that focuses on using Artificial Neural Networks to **approximate functions**



To train a NN we use some algorithm (backpropagation, etc.) to fit the function to labeled data.

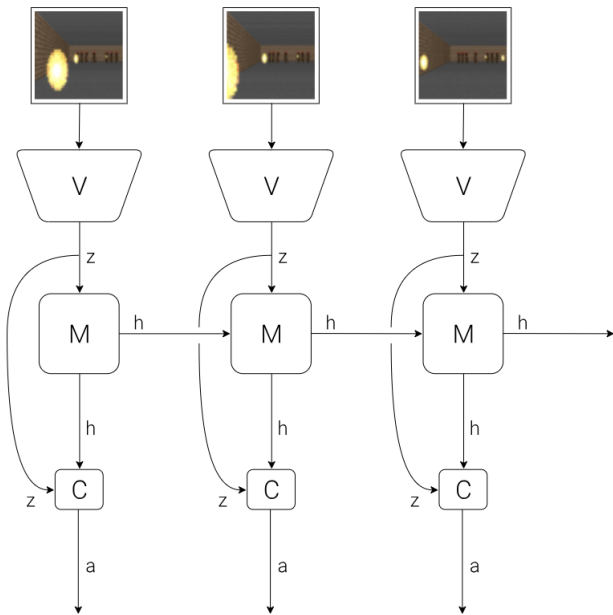
Recall RL's main goal was to select an action that maximizes the expected reward given the current state. Initially we kept a table full of (state, action) values that we could reference to select the best known action.

- What if we used a NN to approximate the "Value State" function
- I.e. input a current state into a neural network to get out an action to take
- State \Rightarrow Network \Rightarrow Action

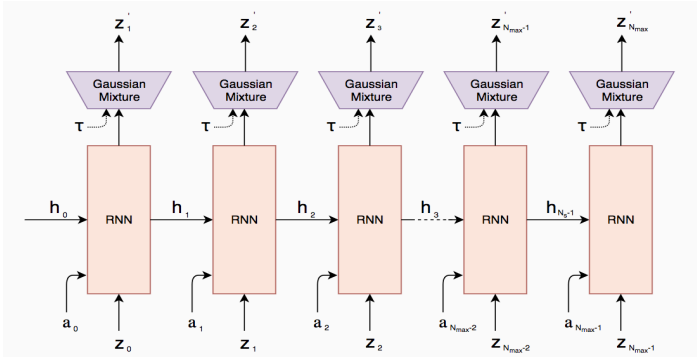
- More generally an agent needs a policy function that maps the current state to the best action, in deep learning this policy function is generally represented by a NN
- BUT this is quite hard to do given that labeled data is not available and learning good state values in DRL takes a huge amount of time and resource compute
- So separate representing (Visual), learning (Model), and acting (Controller) into three different NN...

Architecture separated into 3 parts;

- The (V) Visual Component
 - Takes high dimensional input to latent representation
- The (M) Model Component
 - Makes predictions about the future based on historical information
- The (C) Controller Component
 - Makes decisions given (V) and (M)



- A **Recurrent NN** was utilized to compress what happens over time and predict the future
- LSTM Cells
- Mixture Density Network (MDN-RNN)
 - Trains the RNN to output the probability distribution of the next latent vector z_{t+1} given the current and past information made available to it
 - $P(z_{t+1} | a_t, z_t, h_t)$



- The controller is used to select which action to take to maximize expected reward
- In this paper (C) is a simple linear layer
 - $a_t = W_c[z_t h_t] + b_c$
- (C) is trained using Covariance-Matrix Adaption Evolution Strategy (optimized on single machine CPU)

Deep
Reinforcement
Learning
— and —
Recurrent
World Models
Facilitate
Policy
Evolution

Chris Larsen

Introduction

Deep Learning

Deep
Reinforcement
Learning

Agent Model

Visual
Component
Model
Component
Controller
Component

**Car Racing
Experiment**

VizDoom
Experiment

τ
Cheating

Discussion



1. Collect 10,000 rollouts from a random policy.
2. Train VAE (V) to encode each frame into a latent vector $z \in \mathcal{R}^{32}$.
3. Train MDN-RNN (M) to model $P(z_{t+1} \mid a_t, z_t, h_t)$.
4. Define Controller (C) as $a_t = W_c [z_t \ h_t] + b_c$.
5. Use CMA-ES to solve for a W_c and b_c that maximizes the expected cumulative reward.

Model	Parameter Count
VAE	4,348,547
MDN-RNN	422,368
Controller	867

Method	Average Score over 100 Random Tracks
DQN [53]	343 \pm 18
A3C (continuous) [52]	591 \pm 45
A3C (discrete) [51]	652 \pm 10
ceobillionaire's algorithm (unpublished) [47]	838 \pm 11
V model only, z input	632 \pm 251
V model only, z input with a hidden layer	788 \pm 141
Full World Model, z and h	906 \pm 21

Deep Reinforcement Learning — and — Recurrent World Models Facilitate Policy Evolution

Chris Larsen

Introduction

Deep Learning

Deep Reinforcement Learning

Agent Model

Visual Component
Model Component
Controller Component

Car Racing Experiment

VizDoom Experiment

Cheating

Discussion



1. Collect 10,000 rollouts from a random policy.
2. Train VAE (V) to encode each frame into a latent vector $z \in \mathcal{R}^{64}$, and use V to convert the images collected from (1) into the latent space representation.
3. Train MDN-RNN (M) to model $P(z_{t+1}, d_{t+1} \mid a_t, z_t, h_t)$.
4. Define Controller (C) as $a_t = W_c [z_t \ h_t]$.
5. Use CMA-ES to solve for a W_c that maximizes the expected survival time inside the virtual environment.
6. Use learned policy from (5) on actual Gym environment.

Model	Parameter Count
VAE	4,446,915
MDN-RNN	1,678,785
Controller	1,088

- The RNN model learns things such as game logic, enemy behavior, physics, and graphic rendering
- The temperature can be increased to increase model uncertainty thus making the game even harder than the actual environment

Temperature	Score in Virtual Environment	Score in Actual Environment
0.10	2086 \pm 140	193 \pm 58
0.50	2060 \pm 277	196 \pm 50
1.00	1145 \pm 690	868 \pm 511
1.15	918 \pm 546	1092 \pm 556
1.30	732 \pm 269	753 \pm 139
Random Policy Baseline	N/A	210 \pm 108
Gym Leaderboard ^[34]	N/A	820 \pm 58

- Since the dream environment is governed by (M) the agent may find adversarial policies that exploit the uncertainty of the dream
- Some trajectories through (M) may not follow the governing laws of the actual environment
- (C) has access to h_z of (M)...

- Partially observable environments?
 - Would need to use some sort of iterative training approach
- Train (V) and (M) using backprop on GPU, let (C) be trained with ES on CPU
——
- Could (M) represent a family of environments from some distribution and if so how general could (M) become?
 - Could priors be formed by meta-learning (M) across different settings?
 - The (M) of life!

- Learning the (M) is computationally expensive and requires good data samples
- Thus MAS could utilize a shared model of the environment
 - Allowing the agents to "share" prior information and dynamics of the environment
 - While also allowing each agent to act from its own controller network



Deep
Reinforcement
Learning
— and —
Recurrent
World Models
Facilitate
Policy
Evolution

Chris Larsen

Introduction

Deep Learning

Deep
Reinforcement
Learning

Agent Model

Visual
Component
Model
Component
Controller
Component

Car Racing
Experiment

VizDoom
Experiment

τ
Cheating

Discussion

- David and Schmidhuber, Recurrent World Models Facilitate Policy Evolution, 04-Sep-2018. Available: <https://arxiv.org/abs/1809.01999>.
- <https://worldmodels.github.io/>